DETECTION OF PPE KIT FOR INDUSTRY WORKERS

Srikar Mukkamala, Elavartha Nikhil Reddy, Boppudi Sai Ashrith, Ankadala Hemanth
CSD Major Project
GitHub Link : https://github.com/SrikarMukkamala/ppe_detection_webapp/tree/mlflow

***Abstract*—Surveillance Application to verify PPE in industry In this project, we have to process feeds from video surveillance cameras, detect and track workers and alert with name of the person who is not wearing proper "personal protection equipment" for the specified area/zone. Eg. Construction worker, mill worker, miner etc.**

Keywords{PPE, YOLO, MLFlow, Flask, MLOps, Object Detection}

I. Introduction

Factories can be risky if people don't have the right safety gear and training. New computer programs with smarts (AI) are being used more and more in places where safety is important.

This project is building a system to make sure people wear their Personal Protective Equipment (PPE) like goggles, gloves or masks. It uses special cameras to see what people are wearing and who they are. If someone has the right gear, the system acts like a keycard and lets them enter a special area. This could prevent many accidents and injuries every year.

Every year, a shocking number of people - around 350,000 - die from accidents at work, according to the International Labour Office. Additionally, millions more are injured badly enough to miss work for several days.

Good safety practices in the workplace are linked to greater productivity. However, some jobs are riskier than others. In the United States alone, construction sees the most work-related deaths, with over 5,000 fatalities between 2016 and 2020. This is significantly higher than other industries, like agriculture or finance.

Research by the National Institute for Occupational Safety and Health (NIOSH) points out that a quarter of all construction deaths involve head injuries, and most of these (84%) could have been prevented by wearing a hardhat.

Construction sites often involve heavy machinery and trucks, making it crucial for workers to be easily seen to avoid accidents. Wearing high-visibility vests with reflective material can significantly decrease the risk of serious injuries or death.

II. Goal

This project aims to boost safety for workers in high-risk environments. We're building a powerful system that uses image recognition to check if workers are wearing their Personal Protective Equipment (PPE). This system will combine with an siamese neural network for identifying faces, all while running fast and smoothly in real-time for a user-friendly experience.

We'll be focusing on detecting these specific PPE items:
1. Hard Hat
2. Humans
3. Boots

This project explores a new approach to workplace safety by combining facial recognition with object detection for PPE (Personal Protective Equipment). While both technologies are commonly used in monitoring systems separately, this study is one of the first to combine them. This approach allows the system to identify individuals and verify they are wearing the required PPE simultaneously, potentially enhancing safety and security in hazardous workplaces.

The project involved detecting five different PPE items (list them here if provided) along with facial recognition. This combined approach offers a unique contribution to the field of workplace safety.

III. Preliminaries

A. Yolo v9

YOLOv9 is a recent advancement in real-time object detection, aiming to improve upon its predecessor, YOLOv8. It achieves this through innovative techniques like Programmable Gradient Information (PGI) and the Generalized Efficient Layer Aggregation Network (GELAN). PGI tackles the challenge of information loss within the neural network, crucial for accurate object detection. GELAN contributes by enhancing information flow and allowing the network to retain crucial details throughout the detection process. These features, combined with YOLOv9's efficient architecture, contribute to its superior performance in terms of accuracy, speed, and efficiency compared to previous versions. Although not yet as widely adopted as earlier YOLO models, YOLOv9 holds promise for various real-world applications requiring high-performance object detection.

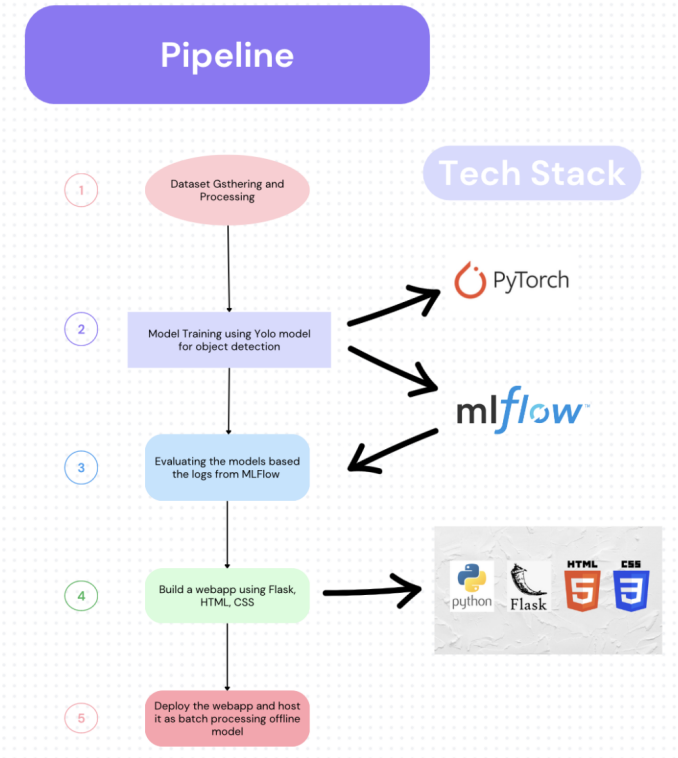
IV. Methodology

A. PPE Detection

This project leverages YOLOv9, a cutting-edge object detection model known for its speed and accuracy, to identify PPE (Personal Protective Equipment) kits. YOLOv9's efficiency makes it suitable for real-time applications. To ensure

accurate detection, we employ a combination of three loss functions: box loss, class loss, and DFL. These functions help the model refine its predictions during training. Additionally, we utilize three metrics: precision, recall, and MAP, to evaluate the model's performance and effectiveness in detecting PPE kits accurately.

B. High Level Design



C. Tech Stack and Life Cycle

.

1. Requirement Gathering :
User should be able to upload video or directly run his webcam on which inference is to be performed.Video and User feed back should be stored.

2. Transforming into ML task :
Leveraging on Deep Learning techniques such as object detection to extract features from the custom dataset and building the ML model would help in detecting the PPE kits.

3. Dataset Gathering :
The Dataset contains 4,380 training images, 420 validation images and 211 testing images along with labels for each image. Appropriate data augmentation such as modifying color space, etc were performed.

4. Model Development :
Developed a Yolo model by training it on the custom dataset to inference it on videos for object detection.

5. Model Evaluation :
Various metrics such as mAP, Precision and Recall were taken into consideration and utilized MLFlow tool for ML model evaluation by accounting different runs, metrics, parameters and artifacts obtained from MLFlow.

6. Model Deployment :
After the best model was obtained with the help of MLFlow, further develop a webapp using Flask for backend and HTML, CSS for frontend and host it on the web.

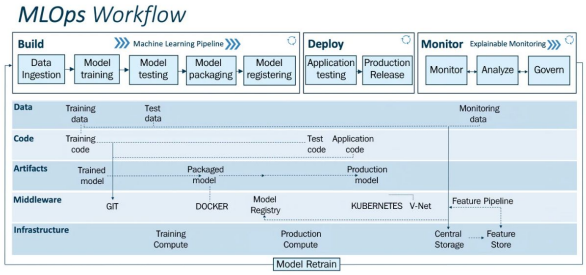
7. Monitoring and Infrastructure :
Batch processing with offline model will be deployed as we do not need real time processing as the data need to be updated immediately. By periodically measuring the model's performance and training it on updated data and performing A/B testing, we ensure model is efficient.

V. Metrics

Latency: Measure the time taken for the system to process and respond to incoming video feeds.
Throughput: Evaluate the number of video feeds processed per unit of time.
Error Rate: Monitor the accuracy of PPE detection and facial recognition to identify potential issues.
Resource Utilization: Track CPU and memory usage to optimize resource allocation and identify bottlenecks.

VI. Operation Strategies

Continuous Monitoring: Implement monitoring tools to track system health, performance, and potential issues in real-time.
Regular Maintenance: Schedule regular maintenance tasks, such as model updates, to ensure the system stays up-to-date and continues to perform effectively.
Incident Response: Develop a robust incident response plan to address unexpected issues promptly and minimize downtime.
Backup and Recovery: Implement regular backup mechanisms to recover from data loss or system failures efficiently.
Deployment and Inference: We will try to implement a forward proxy where we can also have a rate limiter and load balancer where users can be directed to different models according to CI/CD pipeline and Model improvement.



VII. MLFlow

MLflow is an open-source platform for managing the end-to-end machine learning lifecycle. It simplifies the process of tracking experiments, packaging code into reproducible runs, and sharing and deploying models across different platforms. MLFLow mainly include :

.

1.Tracking :

MLflow Tracking allows users to log and query experiments, code, and results. This enables researchers and data scientists to track parameters, metrics, and artifacts (such as models, datasets, and plots) across the entire model development process.

2.Projects :

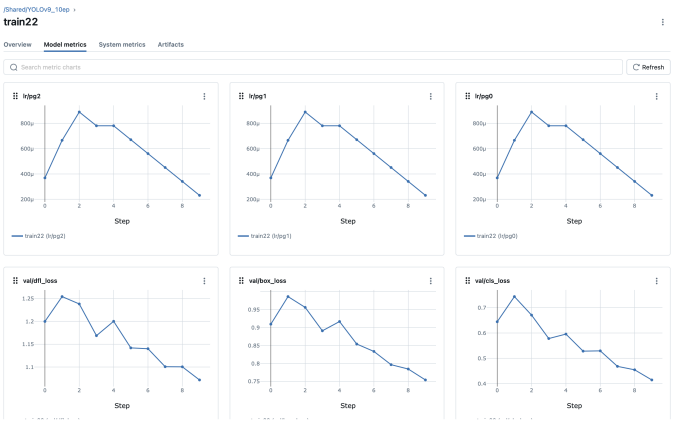
MLflow Projects provide a standard format for organizing and sharing code, data, and environment dependencies required to reproduce a machine learning workflow. This promotes reproducibility and collaboration among team members.

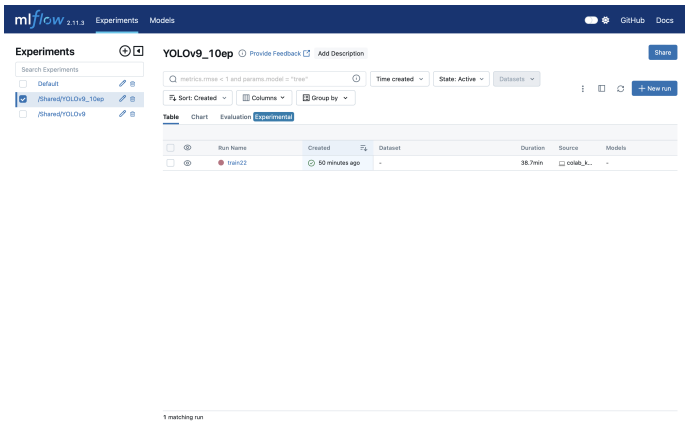
3.Models :

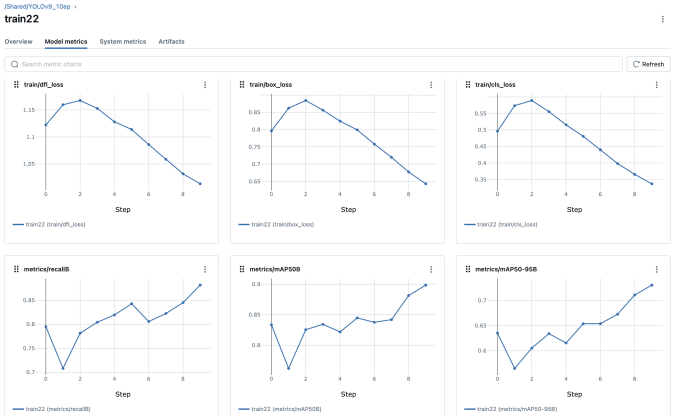
MLflow Models offers a simple way to package and deploy machine learning models in various formats (e.g., Python functions, Docker containers, or Apache Spark UDFs) for use in diverse production environments.

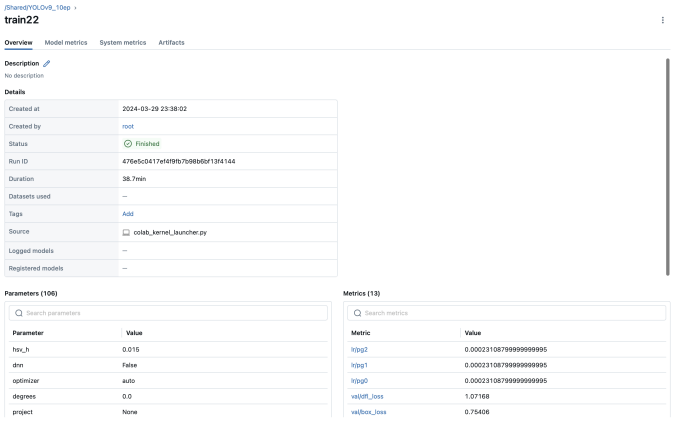
4.Registry:

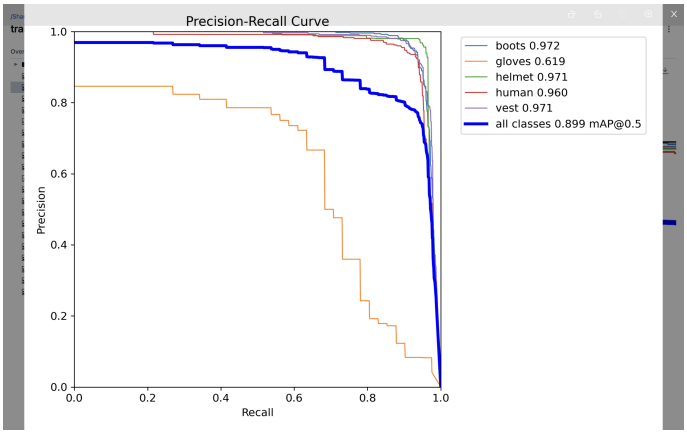
MLflow Registry is a centralized repository for managing model versions, allowing users to track model lineage, compare performance metrics, and easily transition between different model versions.



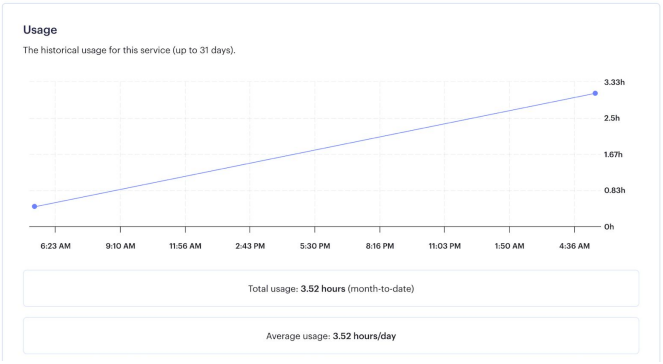


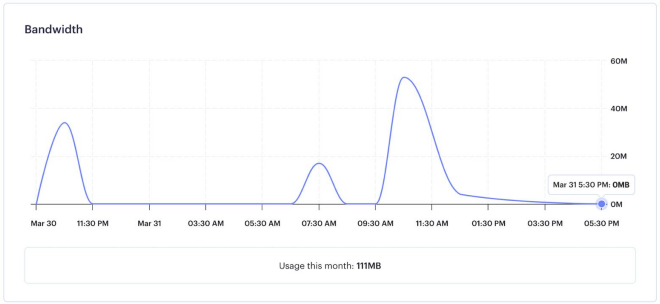


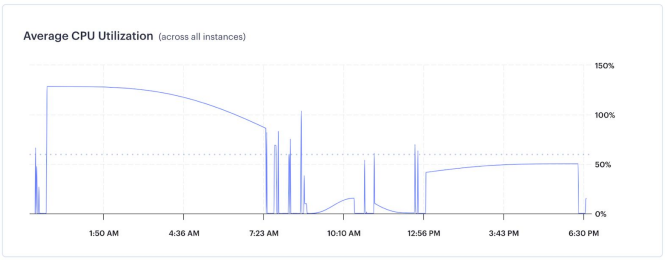




VIII. User Metrics



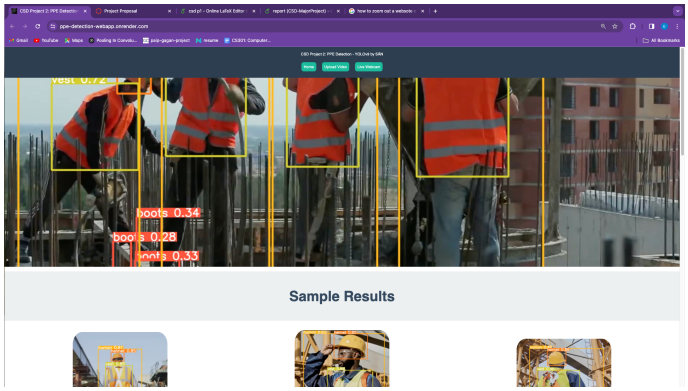


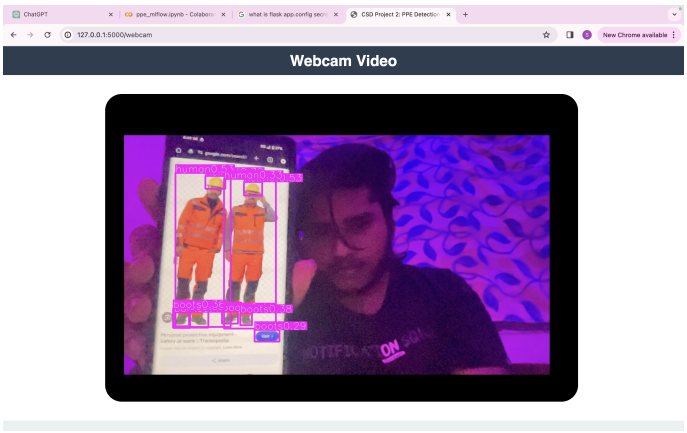


IX. Individual Contribution

Srikar - Model & App Development with MLFlow & Back-end Deployment
Nikhil - Front-end development of the app & Testing
Hemanth - Back-end Deployment & Hosting of the Model
Ashrith - Testing of the Model and rendered webapp

X. Website working





XI. Future Works

1. Fix the bugs of current website.
2. Update the model using new data.
3. Perform A/B testing for target (new) model and control (old) model.

XII. References

1. https://mlflow.org/docs/latest/index.html
2. https://arxiv.org/abs/2402.13616
3. https://docs.render.com/deploys
4. https://flask.palletsprojects.com/en/3.0.x/quickstart
5. https://docs.ultralytics.com